



九十三年度太空科技任務導向 委託研究計畫 (期末報告)

中文計畫名稱：衛星推進器引致污染及衝擊影響分析
軟體發展(2/2)

英文計畫名稱：Development of a DSMC Code for
Analyzing the Contamination and Plume Impingement on
Spacecraft(2/2)

計畫編號：93-NSPO(A)-PC-FA04-01

計畫執行期限：93.07.16~94.07.15

申請機構：國立交通大學機械工程學系

執行單位：國立交通大學機械工程學系

計畫主持人：吳宗信

日期：中華民國 93 年 9 月 15 日

Development of a DSMC Code for Analyzing the Contamination and Plume Impingement on Spacecraft (2/2)

J.-S. Wu, K.-C. Tseng, Y.-M. Lee and Y.-Y. Lian

Department of Mechanical Engineering, National Chiao-Tung University, Taiwan

SUMMARY

The objectives of the current project are to develop an accurate and efficient DSMC code and to apply to the analysis of the plume impingement on spacecraft on PC cluster system. The proposed research is planned for a 2-year period.

In the first year, we have successfully developed a 3-D parallel direct simulation Monte Carlo code using variable-time step method and dynamic load balancing technique to reduce the computational time dramatically. Formosan Sat-3 was used as an example model to demonstrate its capability in analyzing the plume impingement on spacecraft.

In the current report, it is shown to integrate a simulation system including a preprocessor for the input data and DSMC code. At the same time, we helped NSPO to set up a 12-node PC-cluster system. Finally, we implement the developed software on the PC cluster system and train the engineers to use the analysis tool to analyze plume impingement on spacecraft.

KEY WORDS: direct simulation Monte Carlo, parallel, graph partition, dynamic domain decomposition, hypersonic flow, ROCSAT-3, PC clusters, Plume Impingement, Spacecraft, Variable Time Step, Dynamic Load Balancing

TABLE OF CONTENTS

SUMMARY.....	1
TABLE OF CONTENTS.....	2
CHAPTER 1 INTRODUCTION.....	3
CHAPTER 2 DESIGN OF VIISUAL PREPROCESSOR.....	7
2.1 General Description.....	7
2.2 Design Architecture	8
CHAPTER 3 RESULTS AND DISCUSSIONS.....	10
3.1 PDSC	10
3.2 GUI Preprocessor Program for PDSC.....	10
3.3 Installation of the PC-cluster at NSPO	13
CHAPTER 4 CONCLUSIONS	15
REFERENCES	16
APPENDIX A: PDSC User Manual	A1
APPENDIX B: NSPO-PDSC Visual preprocessor User Manual.....	B1
APPENDIX C Notes for Installing Diskless PC-Cluster	C1

CHAPTER 1

INTRODUCTION

DSMC method has become a widely used computational tool for the simulation of gas flows in which molecular effects become important [1]. Detailed reviews of the DSMC method and related computational techniques are thoroughly reviewed in the report [2]; thus, they are skipped here for brevity. One of the direct applications of the DSMC method is to simulate numerically the plume impingement from the reaction control thrusters on satellite, which is otherwise very hard to simulate experimentally due to the cost and technical difficulties. In this project, we proposed to complete the followings in the period of two years: 1) To complete a parallel direct simulation Monte Carlo code [PDSC] that is capable of running on memory-distributed PC-cluster system; 2) To complete a visual preprocessor using VTK[3] and QT[4] toolkits for setting up the boundary conditions and initial domain decomposition; 3) To assist NSPO to set up a PC-cluster system having 12 processors and to install the complete PDSC and visual preprocessor onto the PC-cluster system; 4) To train NSPO personnel to use this system and to provide a set of self-explanatory manual for future reference.

In the *first year* of this two-year project, we have successfully developed and verified a parallel direct simulation Monte Carlo code (PDSC) [2]. Some of the

important features of the PDSC are briefly summarized as follows: 1) parallel implementation using dynamic domain decomposition via multi-level graph-partitioning technique; 2) variable time-step scheme 3) three-dimensional unstructured tetrahedral mesh. Completed PDSC has been used to compute the plume impingement from the ADCS thrusters of Formosa-3 satellite. Details can be found in the final report of the first year [2].

In using this PDSC, several tedious preprocessing steps are required, including setting of boundary conditions, domain decomposition and setting of physical or simulation parameters, to name a few. These steps are usually tedious and time-consuming, which may slow down the analysis or even introduce some unseen but detrimental errors in the preprocessing procedure. Sometimes, it even takes much longer time to preprocess the data than to run the numerical analysis. In order to reduce the risk to a minimum, a complete yet easy-to-use preprocessor is strongly needed. A good preprocessor can shorten very much the time needed in preprocessing step for a numerical analysis as well as the time for training an analysis engineer in learning the use of the simulation software.

By using structured grid or unstructured grid, one can discretize the physical domain. Coordinates of grid points in a structured-grid system can be easily related to

the grid numbers using (i,j,k) indices. For example, the coordinates of neighboring grids can be easily obtained using the geometrical relation in the structured-grid system. However, for unstructured grid, the computational mesh is organized in an unstructured data format, where the information of the neighboring grids can be only obtained through indirect addressing, such as connectivity between cells and nodes in the finite-element analysis. The connectivity information is usually defined when the mesh is generated using some commercial meshing tool, which can be used to derive special grid information. One example that is appropriate in the current project is to convert the connectivity into cell neighbor-identifying information, which is needed in particle tracing if unstructured mesh is employed. Thus, to design a visual preprocessor that is easy to use was proposed in the current 2nd-year of the project to help relieve the tedious preprocessing procedure of the completed PDSC.

In recent years, due to the rapid progress of computer technology, especially the graphical capability of hardware, several used-to-be difficult GUIs (graphical user interface) can be easily designed by employing some toolkits, such as VTK [3] and Qt [4], which package the complicated graphical and geometrical functions into easy-to-use library codes. Thus, it becomes feasible for any novice who is interested in this kind of development for a professional visual preprocessor.

In the current project, it was proposed to integrate a simulation system including a preprocessor for input data and DSMC code. At the same time, we will help NSPO to set up a 12-node PC cluster system. Finally, we will implement the developed software on the PC cluster system and train the engineers to use the analysis tool to analyze plume impingement on spacecraft.

CHAPTER 2

DESIGN OF VIISUAL PREPROCESSOR

2.1 General Description

Preprocessing steps are usually required before actual numerical simulation, especially for large-scale simulation. The preprocessing tasks include meshing, boundary-condition settings, initial-condition settings, and specifying some physical and simulation parameters for the simulation. A typical procedure of preprocessing the data for the current PDSC is shown schematically in Fig. 1. Indeed, it is rather tedious and complicated process, which further justifies the necessity of a preprocessor.

There are many commercial codes for mesh generation, such as HyperMesh, GridGen, Icem CFD *etc.* Grid generation becomes much easier nowadays since these commercial meshing tools can generate the mesh as required. However, especially, settings of boundary conditions for 3D unstructured mesh are often very tedious for practical engineering problems due to the complicated geometry involved. Normal wisdom of overcoming this difficulty is to write a preprocessing code to process the meshing data into the format that the simulation code can accept. This preprocessing code is often case-by-case and requires coding changes from time to time.

Since the PDSC completed in the first year of the project uses 3-D unstructured

mesh for particle collisions and data samplings, the difficulties similar to the above exist. One can greatly reduce the inconvenience to a minimum by using a friendly, easy-to-use visual preprocessor. Thus, a visual preprocessor is designed and currently implemented in this project.

2.2 Design Architecture

Detailed design architecture of the NSPO-PDSC visual preprocessor is schematically shown in **Fig. 2**. It consists of five main categories, including **File**, **Export**, **Solvers**, **Tools**, **View** and **Color**. Details of each category shall be demonstrated in **Chapter 3**. Basically, NSPO-PDSC visual processor is coded in C++ Language using both the toolkits of Qt and VTK to help speed up the time to completion. In short, “Qt” is one of kind GUI libraries produced by Troll Tech. Inc. and a window-like programming on Linux system. Most of the important thing is that it is free on the internet. In addition, “VTK” (Visualization Toolkit) is also a free object-oriented visualized graph library. Programmer just needs to call the VTK library pipeline and then the complicated visual drawing can be completed rather straightforward. Also, Qt and VTK are both open source codes written in C++ code, which we can modify within the framework of C++ compiler environment either under Window-based or Linux-system. All the components presented hereafter are coded

using these two free libraries. In the current project, we code it in the platform of Window-based environment for the time being.

CHAPTER 3

RESULTS AND DISCUSSIONS

3.1 PDSC

We have successfully developed and verified a parallel direct simulation Monte Carlo code (PDSC) [2]. Some of the important features of the PDSC are briefly summarized as follows: 1) parallel implementation using dynamic domain decomposition via multi-level graph-partitioning technique; 2) variable time-step scheme 3) three-dimensional unstructured tetrahedral mesh. Completed PDSC has been used to compute the plume impingement from the ADCS thrusters of Formosa-3 satellite. Details can be found in the final report of the first year [2].

A user manual for PDSC is attached as **appendix A**, it contains the introduction of PDSC, how to install the PDSC on a linux pc-cluster system, coupling PDSC with PAMR and the PDSC running procedures.

3.2 GUI Preprocessor Program for PDSC

We have completed a program of the NSPO-PDSC preprocessor. It can be run smoothly on a standard Window operating system. In the following, we will introduce the example of using the NSPO-PDSC preprocessor program.

Fig. 3 - Fig. 16 show the typical window images captured from the process of running NSPO-PDSC. **Fig. 3** shows how the overall dialog window looks like for initialization. Basically all the five categories mentioned in the above are listed on the top for selection. Sub-window in the left-hand side is reserved for displaying the information of all boundary-condition patches, which is shown graphically in the sub-window at the right-hand side. **Fig. 4** shows the typical window image when loading a mesh file from the local hard disk.

Fig. 5 shows the typical window image for the mesh loaded into the preprocessor. At this stage, only surface mesh of outer boundary is shown. In this example, mesh of the Formosa-3 is used as the demonstration case. It is obviously that the settings of boundary conditions are rather tedious if there is no visual preprocessor.

Fig. 6 shows the typical window image when the boundary faces are split for clarity. Each boundary face has different color assigned to it for clear identification. **Fig. 7** shows the typical window image after the boundary faces are defined. Related information of the boundary conditions is listed in the left-hand side of the window. It is subject to be modified if necessary.

Fig. 8 shows the typical window image when I/O faces of the thruster are zoomed in for setting up the corresponding inflow conditions for DSMC simulation. Note that

the thruster is simplified as a cylindrical tube for simplicity. **Fig. 9** shows the typical window image when I/O faces of the thruster are further zoomed in for better visualization as compared with that in **Fig. 8**.

Fig. 10 shows the typical dialog window image for setting up boundary surfaces properties. In this dialog window, temperature, velocities and diffusion percentage can be set up for correct boundary conditions. **Fig. 11** shows the typical dialog window image for setting partition parameters for initial domain decomposition. There are several types of weighting for graph partitioning tool, including constant weighting, volume weighting and specified weighting as well as the number of processors we choose to simulate the PDSC. **Fig. 12** shows the typical window image of partition for 12 processors for the Formosa-3 satellite after partitioning the mesh in the previous step (**Fig. 11**).

Fig. 13 shows the typical dialog window image for setting up the simulation conditions. For example, magnitude of time step, maximum number of iterations, inflow condition, and so on. **Fig. 14** shows the typical dialog window image for setting up simulation conditions of species data if multispecies are involved. These include the species diameter, temperature exponent, collision parameters, to name a few. If rotational and vibrational degrees of freedom are important to the simulation, then a

dialog window (**Fig. 15**) can then be used to set up the related parameters.

Fig. 16 shows the typical dialog window image for setting up some specific simulation conditions for the PDSC. It may include the type of time step (constant or variable), the type of domain decomposition (static or dynamic). In addition, some other schemes are reserved for future development, such as IP method for low-speed rarefied gas flows.

There is a complete user manual for this NSPO-PDSC visual preprocessor attached as **Appendix B**, which details all the operational procedures of this preprocessor program.

3.3 Installation of the PC-cluster at NSPO

We had helped NSPO to construct a 12-processor PC-cluster system (one server, ten client nodes and one PC for pre-processing/post-processing) with Linux OS , MPI as the communication protocol, and a job queuing system for submitting PDSC program. A simplified procedure for installing the PC-cluster system is attached as **Appendix C**.

We had tested the PDSC program on the new PC-cluster system at NSPO. Results show that the integrated system with PDSC and PC-cluster system works very well.

Simulation conditions for the test run using PDSC on this PC-cluster system include the following:

1. Simulated domain: simplified model of Formosa-3.
2. Simulated particle number: about half million.
3. Processor number: 2, 4, 8 and 10.

Fig 18 illustrates the corresponding surface properties of the Formosa-3, where the physical details are not explored here for brevity. **Figure 17** presents the resulting parallel performance using PDSC on this new PC-cluster. Results show that with the relatively small problem size ($\sim 0.5M$ particles) the parallel speedup can reach 62% at 10 processors, which is reasonably good in general. For a more practical 3D case, the speedup at 10 processors can be expected to reach 80-90% as shown in our previous work [5].

CHAPTER 4

CONCLUSIONS

We have completed the two-year project with the following achievements. In the *first year*, we have successfully developed a 3-D parallel direct simulation Monte Carlo code using variable-time step method and dynamic load balancing technique to reduce the computational time dramatically. Formosan Sat-3 was used as an example model to demonstrate its capability in analyzing the plume impingement on spacecraft.

In the *second year*, we integrated a simulation system including a preprocessor for the input data, DSMC code and a postprocessor for visualization the results. And we have helped NSPO to set up a 12-node PC cluster system. Finally, we implemented the developed software on the PC cluster system and use the analysis tool to analyze plume impingement on spacecraft. We run a test case as a middle size problem, the speedup of 10 CPUs case is about 6.2. According to our experience, the speedup will be better for a larger size case.

The NSPO-PDSC preprocessor is helpful for speeding up several tedious preprocessing steps, including setting of boundary conditions, domain decomposition and setting of physical or simulation parameters, to name a few. It can shorten the preprocessing procedure time without some unseen but detrimental errors.

REFERENCES

1. Bird GA. Molecular Gas Dynamics and the Direct Simulation of Gas Flows.
Oxford University Press, New York, 1994.
2. Final report: Development of a DSMC Code for Analyzing the Contamination and
Plume Impingement on Spacecraft(1/2), project number:
92-NSPO(A)-PC-FA06-01
3. VTK Home page, <http://www.vtk.org/>.
4. Trolltech company home page, <http://www.trolltech.com/>.
5. J.-S. Wu and K.-C. Tseng, "Parallel DSMC Method Using Dynamic Domain
Decomposition," International Journal for Numerical Methods in Engineering, Vol.
63, pp. 37-76, 2005

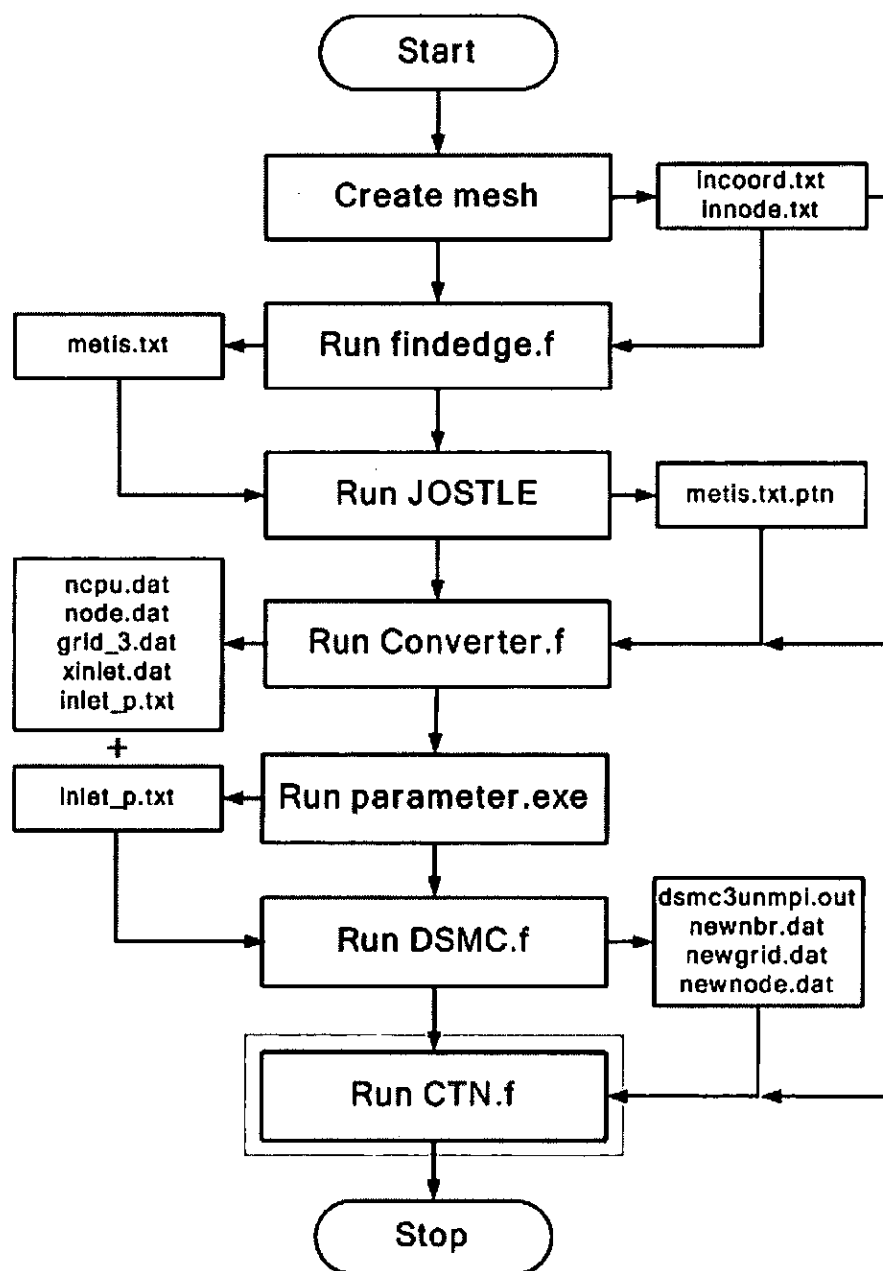


Figure 1. Typical procedure of preprocessing the input data in the PDSC.

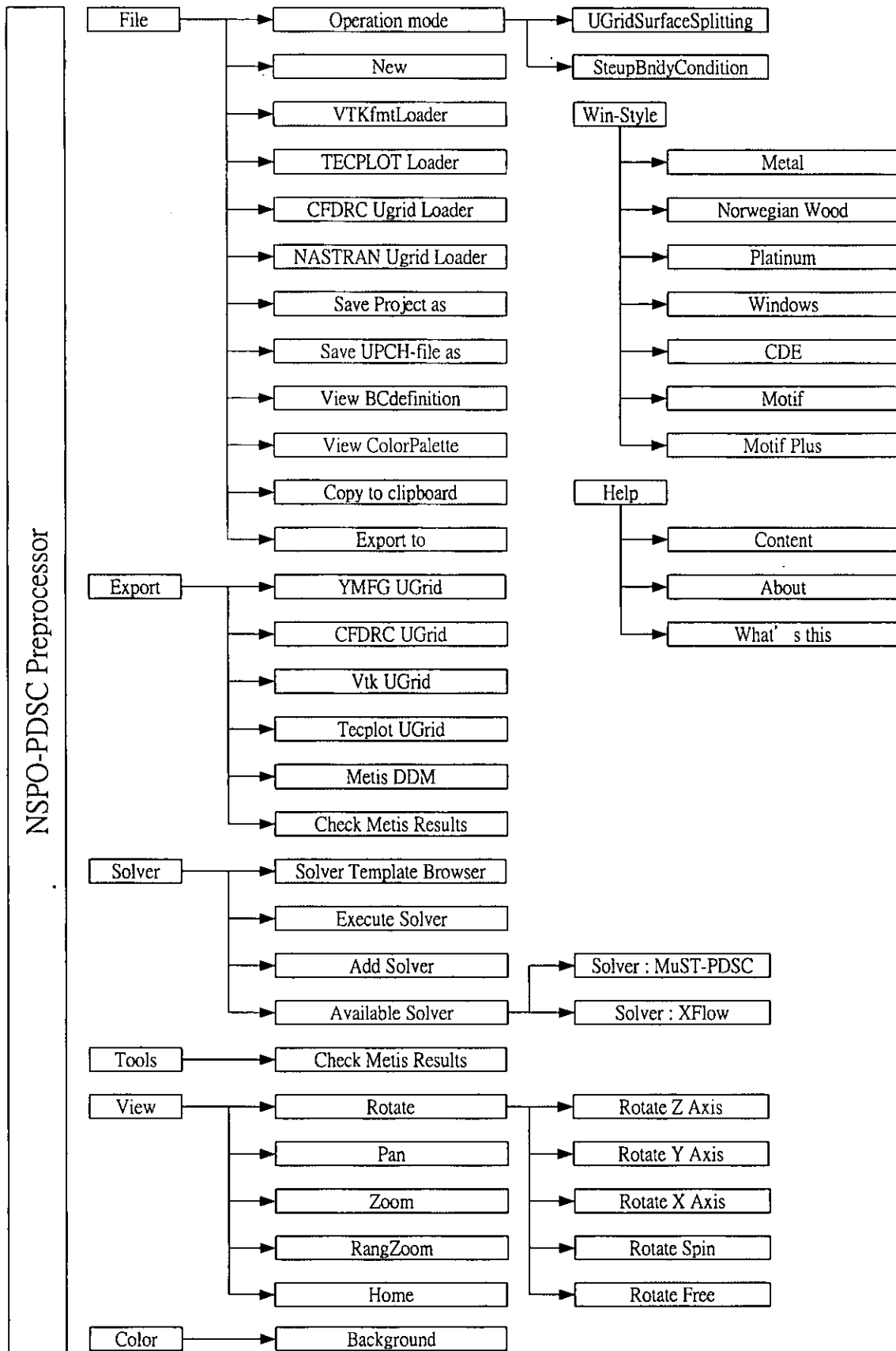


Figure 2. Design architecture of NSPO-PDSC-Preprocessor.

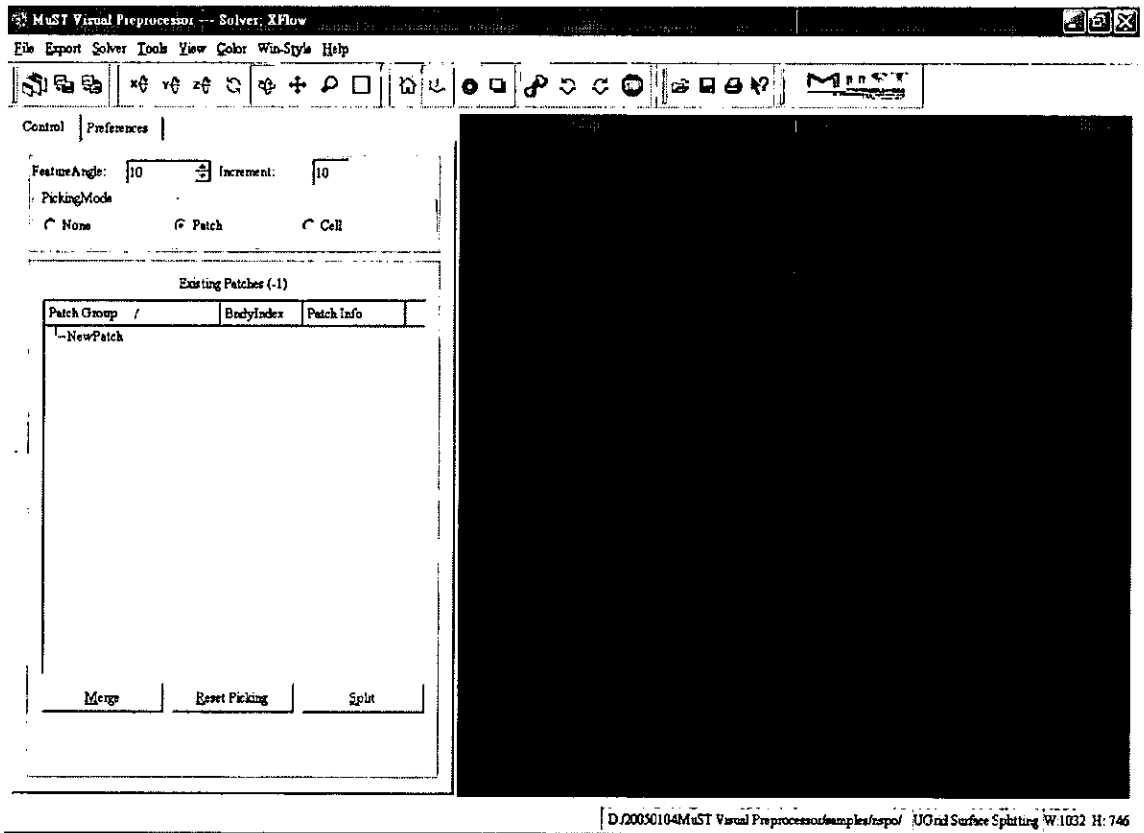


Figure 3. Window for initial loading program dialog

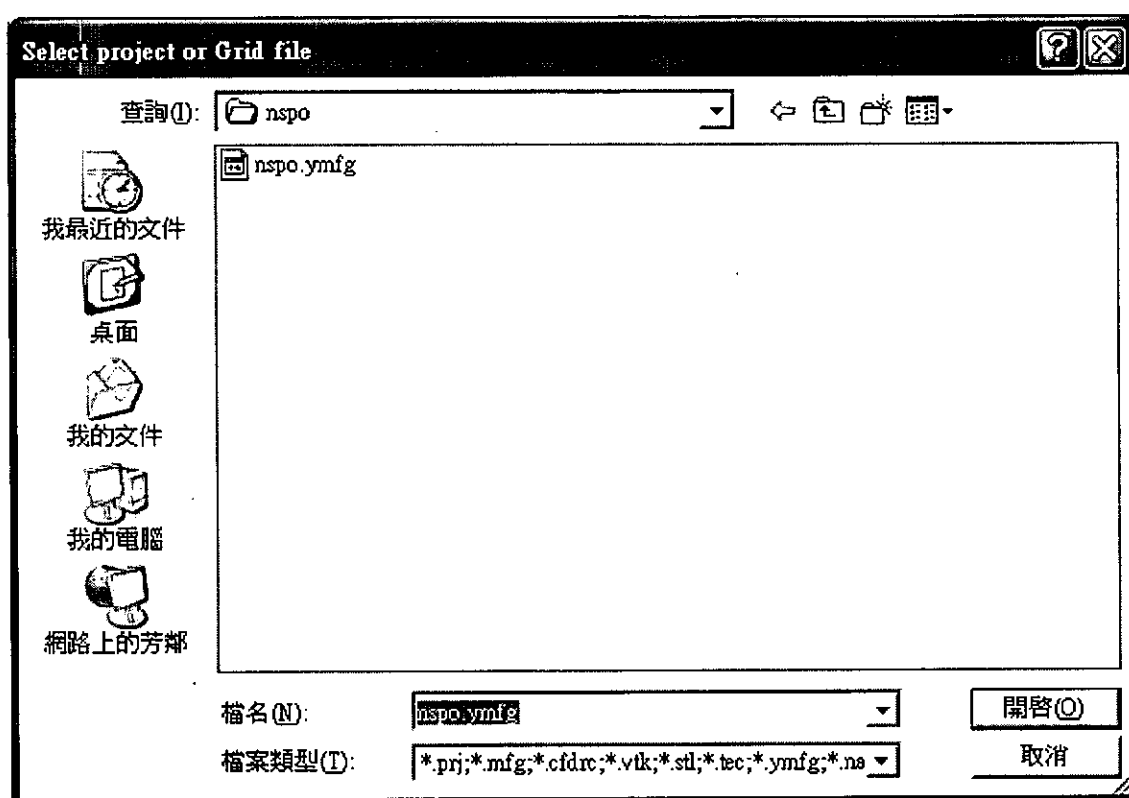


Figure 4. Window for open grid file or project file

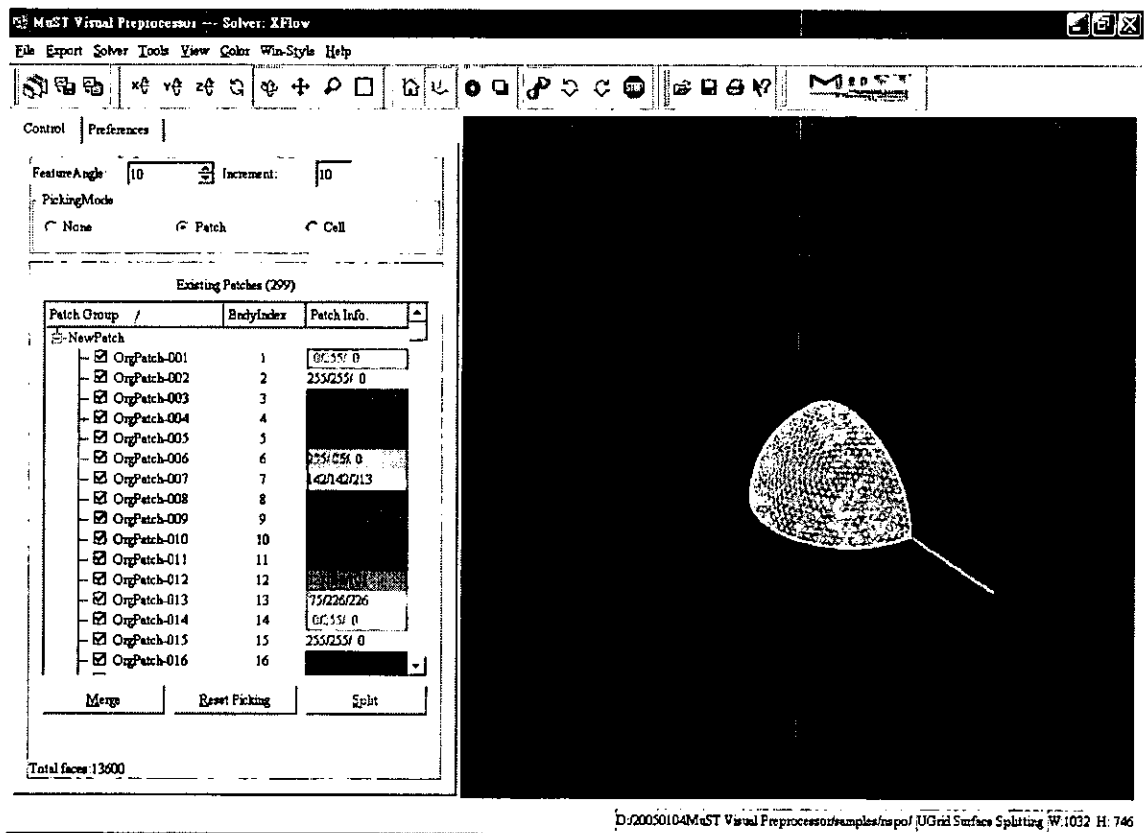


Figure 5. Typical window image for the initial loading of a mesh.

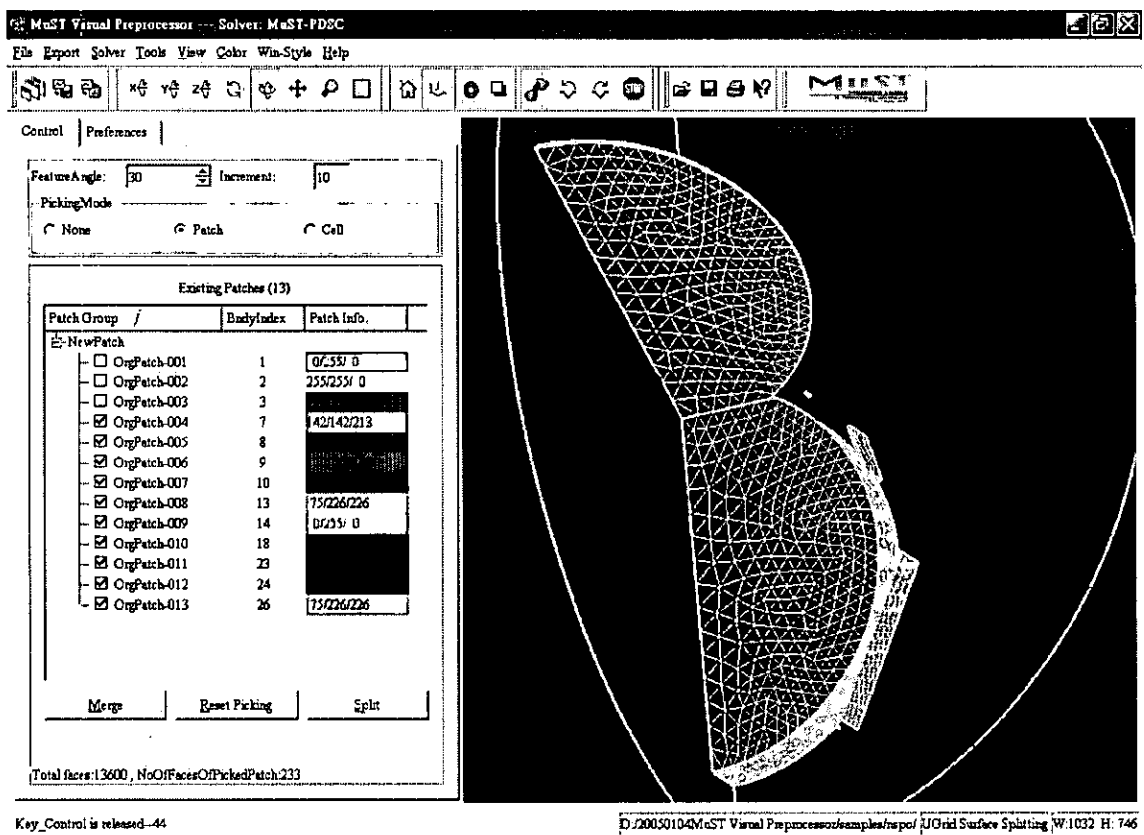


Figure 6. Typical window image when the boundary faces are split.

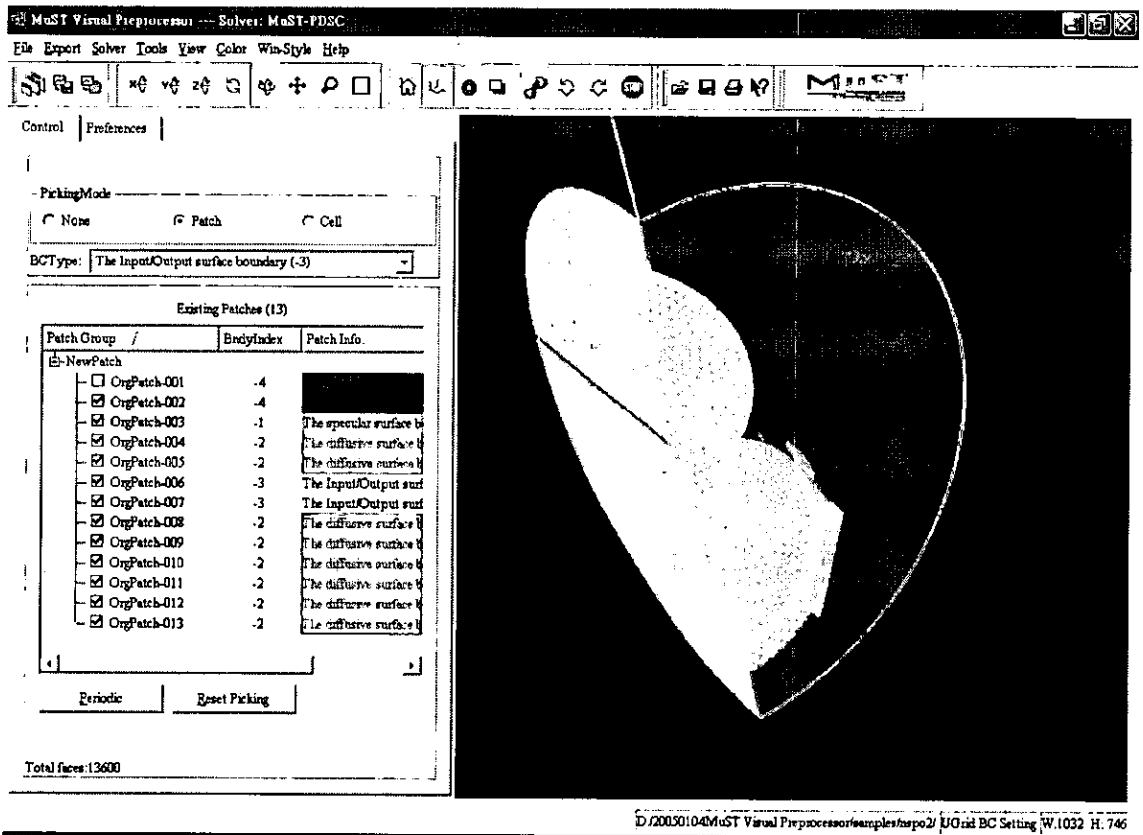


Figure 7. Typical window image after the boundary faces are defined.

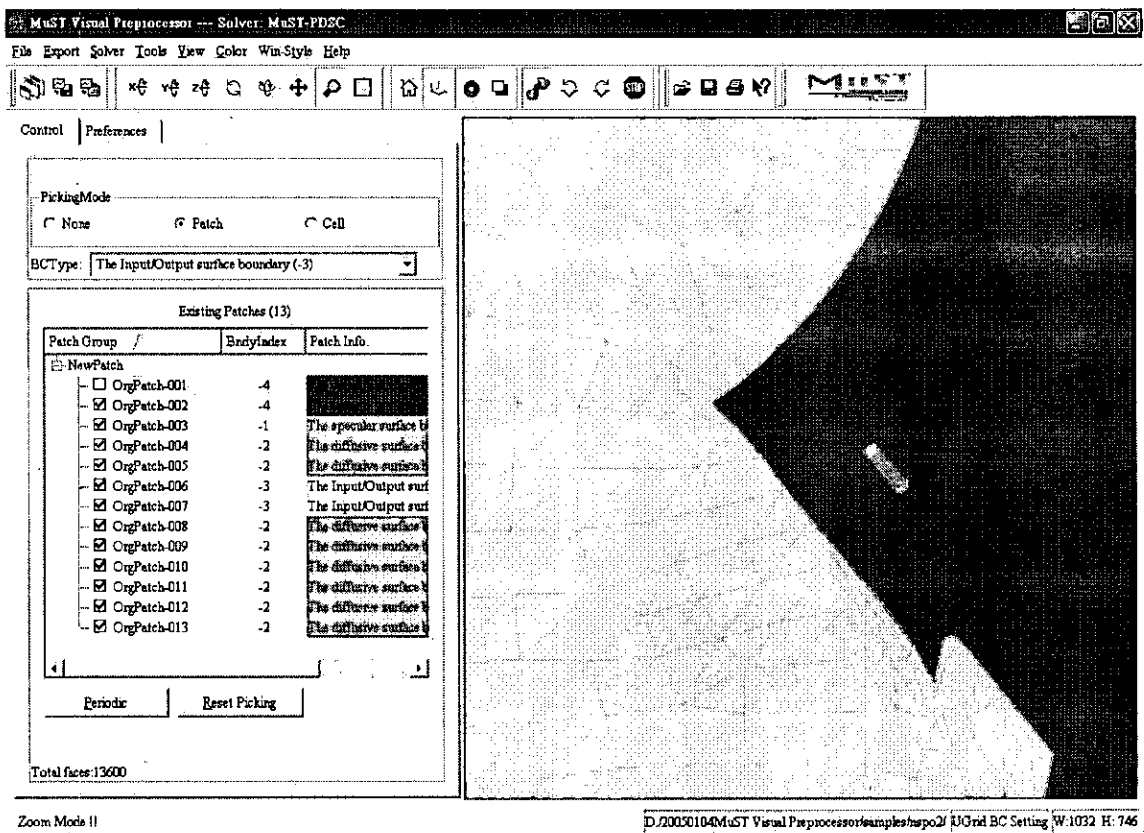


Figure 8. Typical window image when I/O faces of the thruster are zoomed in.

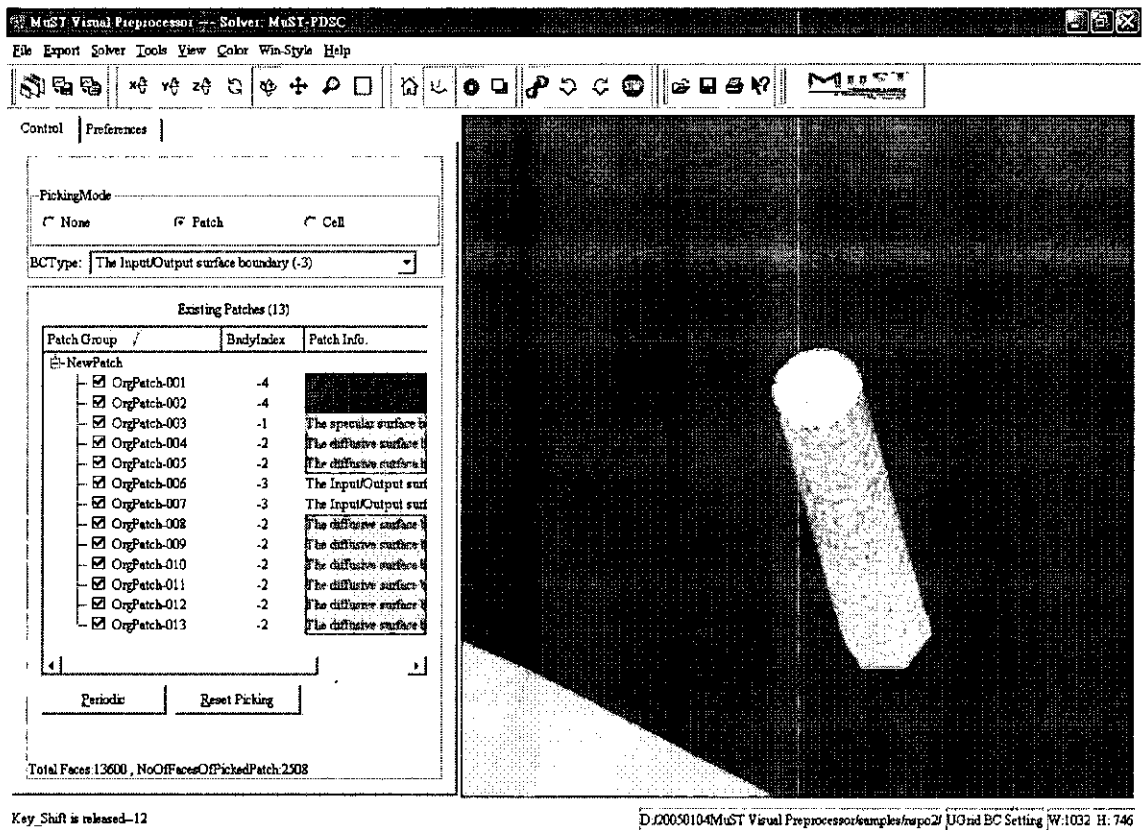


Figure 9. Typical window image when I/O faces of the thruster are further zoomed in as compared with that in Fig. 8.

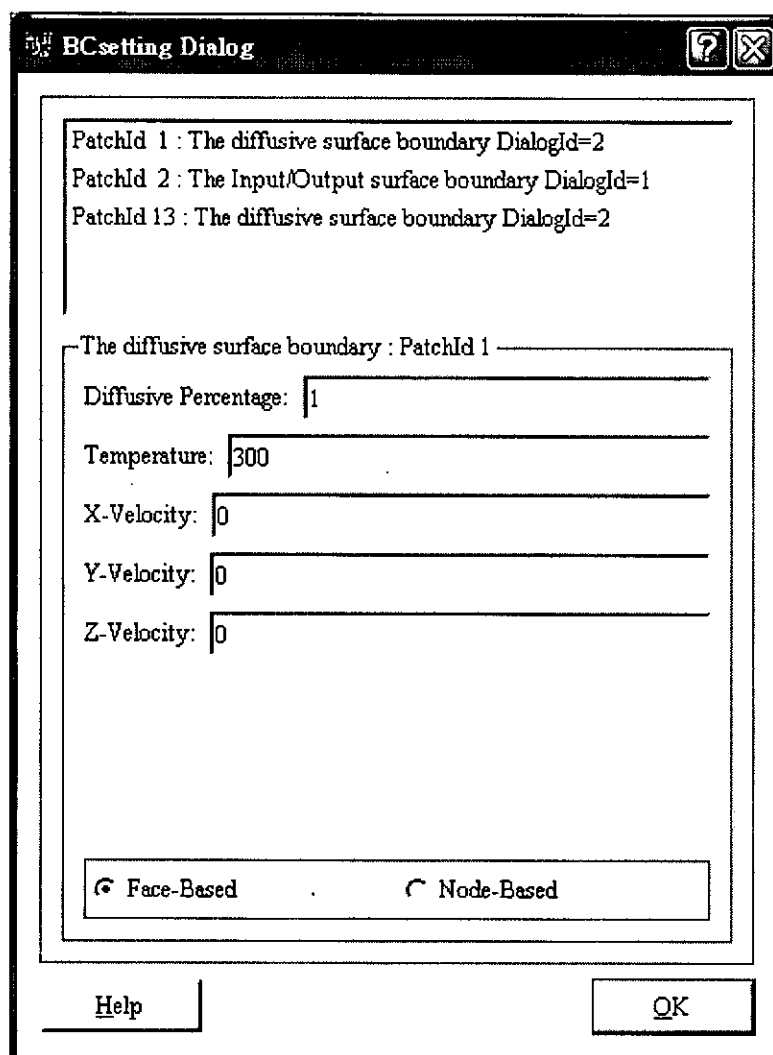


Figure 10. Typical window image for setting up boundary surfaces properties.

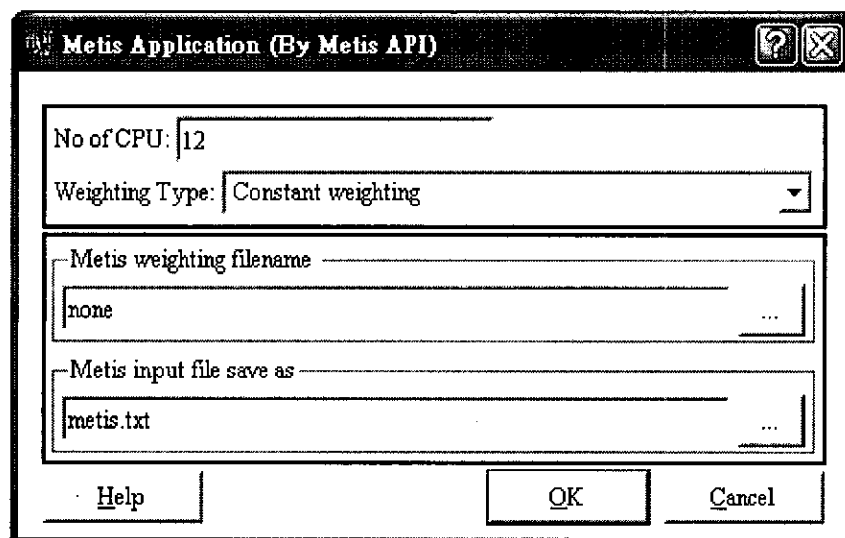


Figure 11. Typical window image for setting partition parameters for initial domain decomposition.

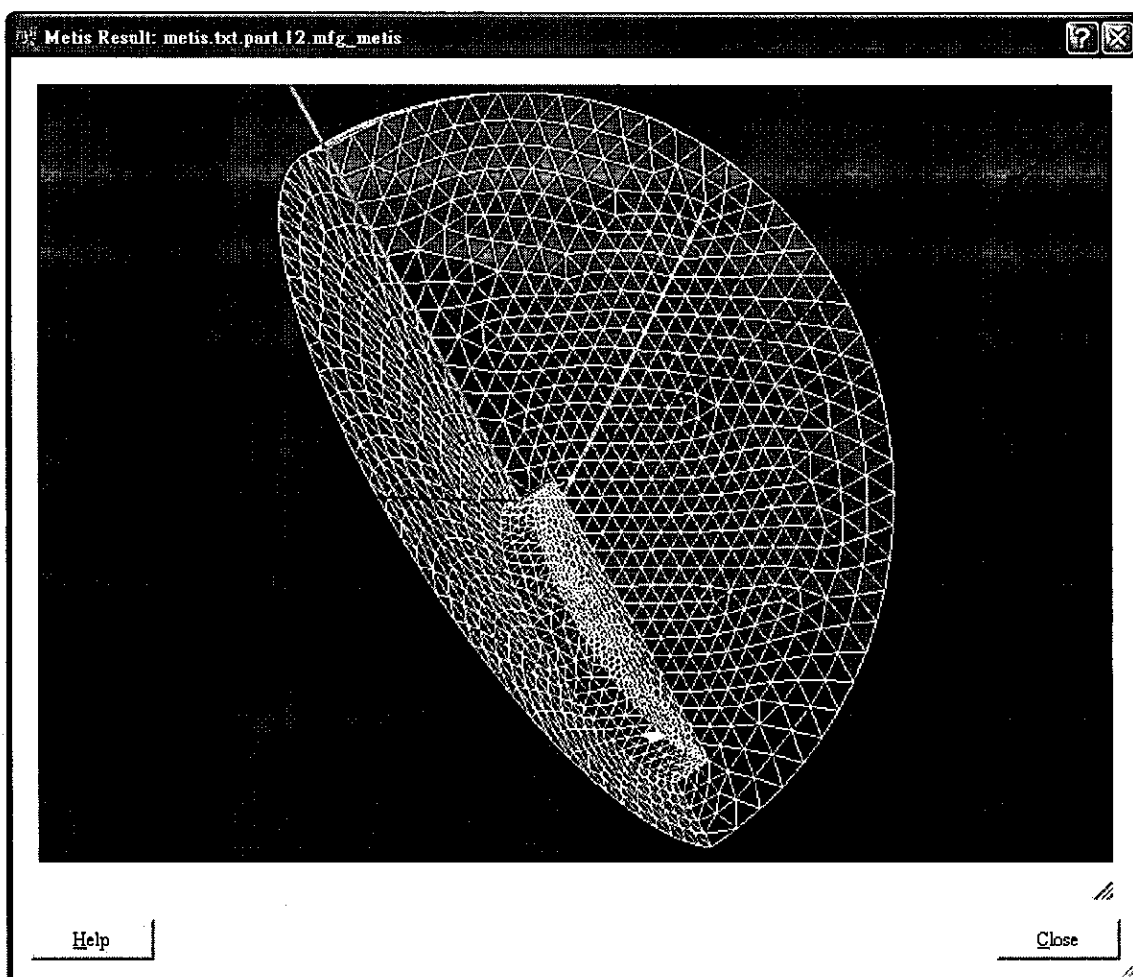


Figure 12. Typical window image for after the partition of the mesh for the Formosa-3 satellite.

Solver control: McST-PDSC

Simulation Conditions | Species Properties | Rotational Vibrational Model | Simulation Modules | Chemical Reaction | Initial Conditions

NMF	4
MNOD	4
MNC	144709
MNCG	144709
NNOD	30320
MM	2000000
NIOBC	20000
NSBF	20000
NIOFACE	20000

Range of simulated domain	
Xmin:	-0.000001
Xmax:	0.015
Ymin:	-0.000001
Ymax:	0.015
Zmin:	-0.000001
Zmax:	0.025
SCALE_RATIO	5.e-3

OMEGA	0
THETA	0
NPBC	10

Stream Options

☐ No Stream

☒ Uniform Stream

Stream Type

☐ Stream+Initial state Vacuum

☒ Uniform Stream

ISPD 0

FTMP 300

FND1 3.219552504e22

FND2 3.219552504e20

Free stream velocity

VFX: 0.0 VFY: 0.0 VFZ: 0.0

WALLP 1

SFTMP 300

Moving surface velocity

SFV(1): 0.0 SFV(2): 0.0 SFV(3): 0.0

NIS 2

NSP 10

NPS 100

NPT 2001

NP 20

NMBG 20000

Help Save as Load OK Cancel

Figure 13. Typical dialog window image for setting up simulation conditions of

PDSC.

Solver control: MuST-PDSC

Simulation Conditions | **Species Properties** | Rotational Vibrational Model | Simulation Modules | Chemical Reaction | Initial Conditions

MNSP 4
MMSG 4

	FSP(i)	ISP(i)	SWEI(i)	SP(1,i)	SP(2,i)	SP(3,i)	SP(4,i)	SP(5,i)
1	0.0	1	1.0	3.003E-10	273	0.74	1.0	
2	0.0	2	1.0	2.92E-10	273	0.67	1.0	
3	0.01	3	1000	4.08E-10	273	0.78	1.0	
4	0.99	4	99000	2.92E-10	273	0.67	1.0	

FSP(i): Fraction of initial particle number of each species
ISP(i): Species belongs to which group for collision
SP(1,i): Reference diameter
SP(2,i): Molecular parameters
SP(3,i): Molecular parameters
SP(4,i): Reciprocal of the VSS scattering parameter (1 is for VHS)
SP(5,i): Molecular mass

Help Save as Load OK Cancel

Figure 14. Typical dialog window image for setting up species data of simulation conditions.

Solver control: MxSY-PDSC

Simulation Conditions

Species Properties

Rotational, Vibrational Model

Simulation Modules

Chemical Reaction

Initial Conditions

Vibrational energy exchange of L-B method

☐ Non-vibrational model
 ☒ Classical vibrational model
 ☐ Quantum vibrational model

Rotational Information for Each Species

4

	ISPR(i)	IZR(i)
1	3	2
2	2	2
3	3	2
4	2	2

4

	1	2	3
1	5	5	5
2	5	5	5
3	5	5	5
4	5	5	5

Vibrational Information for Each Species

Note: i = 1 to MNRP

Note: i = 1 to MNRP

Row: 4

	ISPV(i)	EVIB(i)	IZ' ^
1	1	4	
2	0	0	
3	1	8	

Note: i = 1 to MNRP

Row: 4

	SPV(i)
1	
2	
3	

Table of SPVM(1,i,j) Note: i,j = 1 to MNRP

Column: 4 Row: 4

	1	2	3	4
1	10	10	10	
2	10	10	10	
3	10	10	10	

Table of SPVM(2,i,j) Note: i,j = 1 to MNRP

Column: 4 Row: 4

	1	2	3	4
1	100	100	100	
2	100	100	100	
3	100	100	100	

Help

Save as

Load

OK

Cancel

Figure 15. Typical dialog window image for setting up rotational and vibrational model.

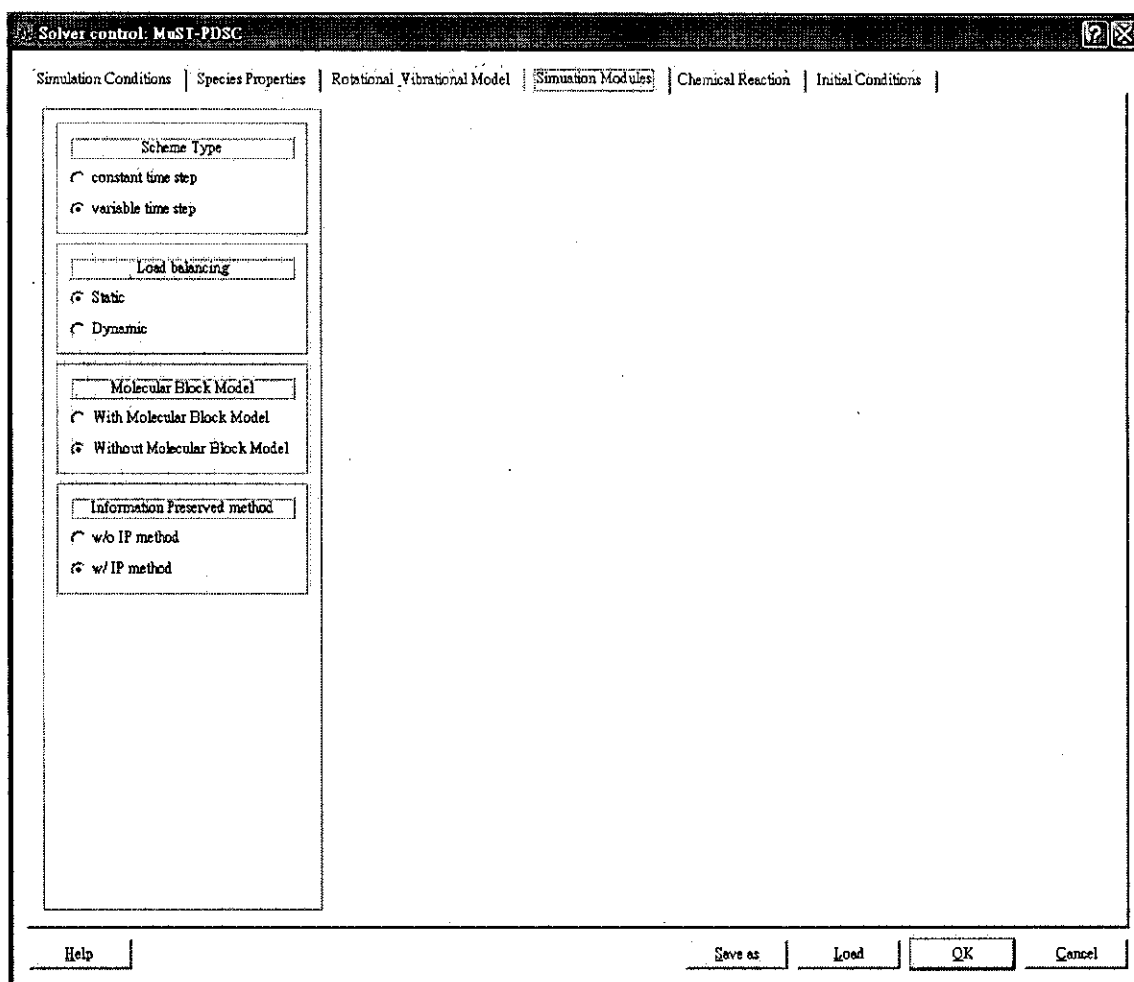


Figure 16. Typical dialog window image for setting up some specific simulation conditions for the PDSC.

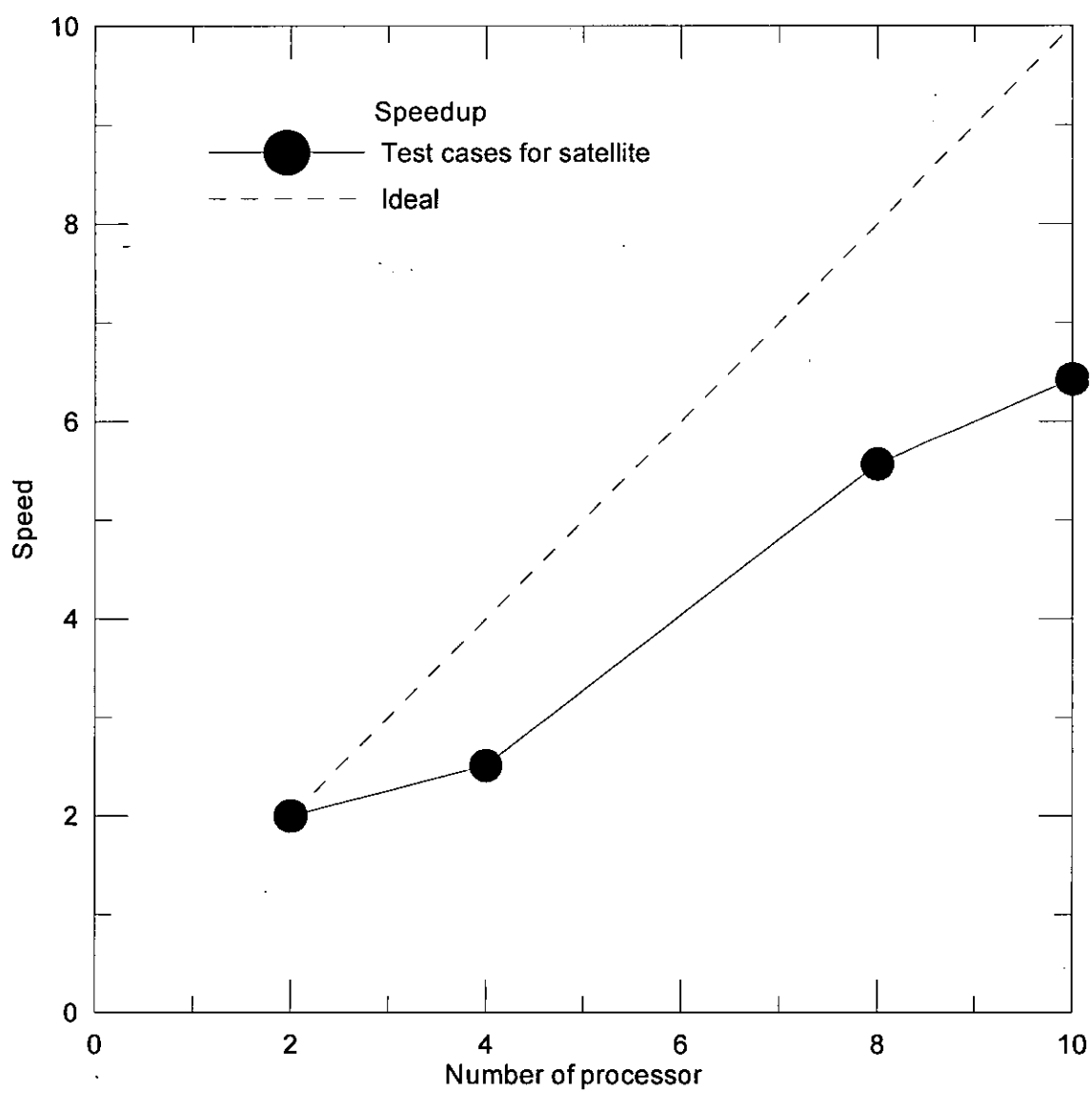
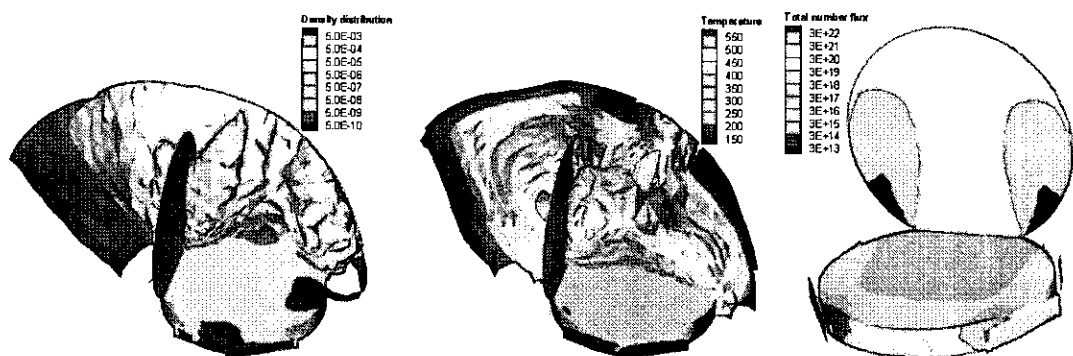


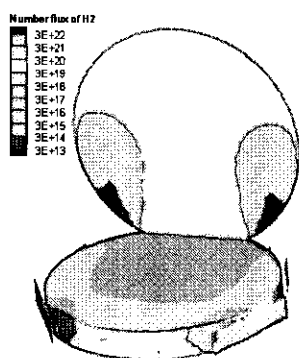
Figure 17. Speedup test



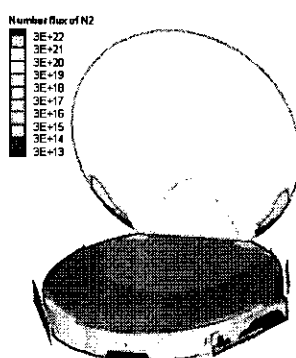
(a) Density distributions

(b) Temperature profile

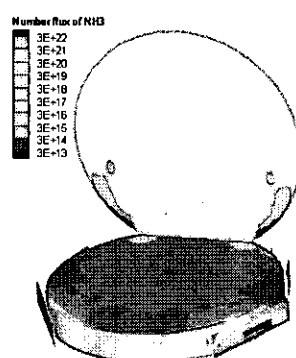
(c) Total number flux



(d) Number flux of H_2



(e) Number flux of N_2



(f) Number flux of NH_3

Figure. 18. Surface properties the test case

APPENDIX B:

NSPO-PDSC Visual preprocessor User Manual

APPENDIX C

Notes for Installing Diskless PC-Cluster

Notes for Installing Diskless PC-Cluster

Diskless nodes do not have any hard-disk, floppy drives and tape drives. They offer significant savings in "Total Cost of Ownership" by eliminating the maintenance costs. Development in diskless PC-cluster with many Linux distributions is relatively popular and is easy to build up for specific computational purposes, for example computational fluid dynamics and rarefied gas dynamics simulations.

1. Hardware

For the current PC-cluster, the Giga-bit Ethernet is chosen as the network system to allow each machine to communicate with each other. The server machine where more hard disks are installed is supposed to provide the required information of diskless system for each of diskless node. The hardware components installed on this server are listed as follows:

- A. Basic PC hardware components: CPU, motherboard, graphic card and 1 GB ram
- B. Giga-bit Ethernet network card
- C. Hard disks: For installing the Linux operation system (Fedora core 3), the service packages of diskless system, the computational packages and storing the results of the computational simulations.

The hardware components installed in each client (computational node) are similar (A. and B.) to the server except the hard disks. Each machine connects to the Giga-bit Ethernet switch hub with cable, there is an extra cable for server machine for the connection to internet.

2. Network sketch

Server has two IP address, and one for intranet while the other is connected to internet. Each client machine has one IP address for intranet.

Recommend: IP address and hostname settings

Item \ machine	SERVER	CLIENTS
Intranet IP	192.168.1.251	192.168.1.1~*
HOSTNAME	server	node001~*

Others network settings:

NIS domain name: nspocluster

3. Operation system installation

Install the Fedora core 3 (FC3) on server machine. It is better to have two or more hard-disks in server machine. One is used to boot and stores the system data, while the other is used for storing the program data. The recommend disk partitions are as follow:

Disk Partitions:

Boot disk :

Mount point	Size
/	Software are installed here. Recommend: >10,000MB
/boot	Boot data. Recommend:120 MB
/home	Users home directory As large as possible.
/tftp	Depends on how many clients, each client takes about 1000MB. Recommend:1000*client number MB

Data disk :

Mount point	Size
/ work	all space in this disk

Note:

Disable the SELinux option, set the network as network sketch, confirms the nis, nfs, dhcp, rsh and tftp service packages are installed.

4. System services for server

There are some important system services of diskless cluster system, as follows:

- A. NIS is a service for account management, accounts are created on server and the clients can acquire accounts data for the server.
- B. NFS is a service for sharing storage space form server machine.
- C. DHCP is a service for assigning the IP address to the client machines.
- D. RSH is a service for remote logging.
- E. TFTP is a service for anonymous transferring booting image file for client machines.

5. Set diskless clients

The Diskless Remote Boot in Linux (DRBL) package is developed by National Center for High-performance Computing (NCHC), the DRBL package is used for constructing a diskless boot environment for Linux. Before starting the setup the environment of diskless system for clients, please check the following notes:

- A. Make sure the server machine is connected to internet.
- B. Set the NIS domain name.
- C. Turn on all services as above mentioned.

The Diskless Remote Boot in Linux (DRBL) package is developed by National Center for High-performance Computing (NCHC), the DRBL package is used for constructing a diskless boot environment for Linux.

Reference to the web site <http://drbl.nchc.org.tw/>, install the DRBL package. When installing the DRBL package, make sure the server is connecting to the internet. After install DRBL package, turn on power of the client machine to check the diskless booting is working or not. Then turn on the RSH service on the client.

6. Computing software

Install the fortran or other computer language compiler.

Install the mpi library for parallel computing, for example: MPICH.

Install the jobs management system.